2

SMOOTHING OF SCATTERPLOTS

JEROME H. FRIEDMAN

WERNER STUETZLE

ORION 003

JULY 1982

Project ORION

Department of Statistics
Stanford University
Stanford, California

SMOOTHING OF SCATTERPLOTS

by

Jerome H. Friedman

and

Werner Stuetzle

Stanford University
and
Stanford L.'near Accelerator Center

ORION 003

JULY 1982

PROJECT ORION

Department of Statistics
Stanford University
Stanford, California

SMOOTHING OF SCATTERPLOTS*

Jerome H. Friedman
Stanford Linear Accelertor Center
Stanford, California

and

Werner Stuetzle
Stanford Linear Accel∋rator Center
and
Department of Statistics, Stanford University
Stanford, California

ABSTRACT

A variable span scatterplot smoother based on local linear fits is described. Lo⌐al cross-validation is used to estimate the optimal span as a function o⁵ ͻbscissa value. A re͐ection rule is suggested to make the smoother resistant against outliers. Computationally efficient algorithms makıng us∂ of updating fo⌐mulas and correspondıng FORTRAN subroutines are presented.

*Work supported by the Department of Energy under contracts DE-AC03-76SF00515 and DE-AT03-81-ER10843, by the Office of Naval Research under contract ONR N00014-81-K-0340, and by the U.S. Army Research Office under contract DAAG29-82-K-0056.

1. Introduction

A smoother is a procedure that operates on bivariate data $(x_1,y_1)\ldots(x_n,y_n)$ and produces a decomposition

$$y_i = s(x_i) + r_i, \quad i=1\ldots n. \qquad (1)$$

Here s is a smooth function, often simply called the smooth, and the r_i are residuals. It is possible to formally define what constitutes a smooth function, and to define measures of smoothness, but for our purposes an intuitive notion will be sufficient. Smoothers are used to summarize the association between the predictor variable X and the response Y. It was pointed out by Cleveland ([illegible]) and is a commonly held belief, that when looking at a scatterplot the eye is distracted by the extreme points in the point cloud, i.e., the fuzzy background, and tends to miss structure in the bulk of the data. Augmentation of the plot by a smooth is a possible remedy. More formally, smoothers can be regarded as curve estimators; one assumes that the response was generated by adding random noise to a smooth function:

$$y_i = f(x_i) + \epsilon_i \qquad (2)$$

and considers the smooth s as an estimate for f.

Recently scatterplot smoothers have found a new use in multiple nonparametric regression (Friedman and Stuetzle, 1981). Let $(x_1,y_1)\ldots(x_n,y_n)$ denote the observations; x_i here is a vector in R^p, not just a single number. Assume as above that $y_i = f(x_i) + \epsilon_i$, $i = 1\ldots n$. Projection pursuit regression constructs an estimate m for f of the form

$$m(x) = \sum_{i=1}^{M} s_i(\alpha_i . x),$$

where the α_i and suitably chosen unit vectors in R^p. For given α_i, s_i (essentially) is found by smoothing the scatterplot of the residuals $r_j^{i-1} = y_j - \sum_{k=1}^{i-1} s_k(\alpha_k . x_j)$ versus $\alpha_i . x_j$. The smoother described in this report is, up to minor modifications, the one used in the current projection pursuit regression procedure.

2. Basic Concepts

According to the definition above, any procedure that passes a smooth curve through a scatterplot, for example a procedure that fits a least squares straight line, would be called a smoother. This is not quite what we have in mind. Assume the data are generated according to (2). We are interested in procedures that can approximate f arbitrarily closely, given a dense enough sample, without any conditions on f apart from f being smooth. Such procedures can be based on local averaging. Take $s(x_i)$ to be the average of the responses for those observations with predictor values in a neighborhood N of x_i:

$$s(x_i) = \text{ave}(y_j | x_j \in N). \qquad (3)$$

Here "ave" can stand for the arithmetic mean, the median, or more complicated ways of averaging to be discussed below. A critical parameter to be chosen is the SPAN, the size of the neighborhood over which averaging takes place. It controls the smoothness of s. The bigger the span, the smother s will be. To obtain consistency, i.e., to make sure that s gets arbitrarily close to f as the sampling rate increases, one must shrink the diameter of the neighborhood in such a way that the number of observations in the neighborhood still grows to infinity. Shrinking the neighborhood makes the systematic or bias component in the estimation error diminish, while increasing the neighborhood sample size guarantees that the variance component of the error goes to zero as well.

An alternative method for nonparametric curve estimation is based on series expansions: make an ansatz for s of the form

$$s(x_k) = \sum_{i=1}^{M} \alpha_i g_i(x_k)$$

where the $g_i(x)$ can, for example, be polynomials or trigonometric functions. The constants α_i are then determined by fitting the series to the data, most commonly by least squares. The role of the span is played here by M, the number of terms included in the model. Trigonometric functions have been used with success in cases where the signal is naturally periodic. If the abscissas x_i are equi-spaced, the fit is particularly inexpensive to compute using the Fast Fourier Transform. Both conditions are usually not fulfilled in the case of scatterplot smoothing, making the method less attractive. The use of polynomials has the drawback that they are not well suited to represent a wide variety of commonly encountered functions, for example, functions with asymptotes.

There are, of course, connections between smoothing by series expansion and smoothing by local averaging. If the series is fitted by least squares, the fitted values $s(x_i)$ are weighted averages of the responses y_i. Depending on the abscissas and the functions $g_i(x)$, the weights determining $s(x_i)$ might or might not be concentrated on responses with corresponding predictor values close to x_i. If they are, the series expansion method behaves like a local averaging method. An example of this is least squares fit of cubic splines which will be further discussed in Section 9.

3. A Simple Nonresistant Smoother

The simplest example for a local averaging type smoother is the moving average, where "ave" in equation (3) denotes the arithmetic mean. The size of the neighborhood is usually specified by the span, the number k of observations to be included in the averaging. We will assume k to be odd and the abscissas x_i to be in increasing order. The neighborhood can be chosen either symmetrically, containing k/2 observations to the left of x_i and the same number to the right, or it can be chosen to contain the k nearest neighbors of x_i, including x_i. (We assume that k/2 is computed by integer division.) There are no general results on which of these two possibilities is better statistically. The nearest neighbors approach generalizes to higher dimensions, but the choice of a symmetric neighborhood is computationally simpler in that exactly one point enters and one point leaves the neighborhood as one moves from observation i to observation i+1. We will, in the following, use symmetric neighborhoods. The boundaries, where it is not possible to keep N symmetric, have to be treated specially; a commonly used adjustment is to shrink the neighborhood so that for i=1 and i=n, one averages only over k/2+1 observations. With these conventions, the moving average smoother is defined by

$$s(x_i) = \mathrm{mean}(y_j \,|\, \max(i-k/2,1) \leq j \leq \min(i+k/2,n)).$$

Obviously, the mean does not have to be recomputed every time. It can be updated, reducing the computation from nk to n. Such updating can be done for all the smoothers we will consider, and is highly desirable because in typical applications k is 5% to 30% of n, and thus the savings are substantial. The simple moving average smoother has some serious shortcomings. One disturbing property is that it does not reproduce straight lines if the abscissa values are not equi-spaced. Another disturbing feature is the bad behaviour at the boundaries. If, for example, the slope of the underlying function f is positive at the right boundary, the estimate for observations close to the boundary will be biased downwards; if the slope is negative, the estimate is biased upwards. Both problems can be alleviated by fitting a least squares straight line L to the observations in the neighborhood instead of fitting a constant and taking the value of the line at x_i as the smoothed value. This obviously solves both problems mentioned above. For the computation, again updating formulas can be used. The slope β and intercept α of the least squares straight line through a set of points $(x_1,y_1)\ldots(x_m,y_m)$ are given by

$$\alpha = \bar{y}_m - \beta\bar{x}_m$$

$$\beta = \frac{C_m}{V_m}$$

with

$$\bar{x}_m = \sum x_i/m,$$

$$\bar{y}_m = \sum y_i/m,$$

$$C_m = \sum (x_i-\bar{x}_m)(y_i-\bar{y}_m),$$

$$V_m = \sum (x_i-\bar{x}_m)^2.$$

When we want to add an observation (x_{m+1}, y_{m+1}), we can make use of the following easily derived formulas:

$$\bar{x}_{m+1} = (m\bar{x}_m + x_{m+1}) / (m+1),$$

$$\bar{y}_{m+1} = (m\bar{y}_m + y_{m+1}) / (m+1),$$

$$C_{m+1} = C_m + \frac{m+1}{m}(x_{m+1} - \bar{x}_{m+1})(y_{m+1} - \bar{y}_{m+1}),$$

$$V_{m+1} = V_m + \frac{m+1}{m}(x_{m+1} - \bar{x}_{m+1})^2.$$

Analogous formulas can obviously be used for removal of an observation from the set.

4. Choice of Span

The most important choice in the use of a local averaging smoother is the choice of the span value. If the smoother is regarded as a curve estimator, then the span controls the trade off between bias and variance of the estimate. We illustrate this for the case of a simple moving average smoother. In this case, the smoothed value at point x_i is given by

$$s(x_i) = \frac{1}{k} \sum_{i-k/2}^{i+k/2} y_j.$$

If we assume that the errors ϵ_i are i.i.d. with expected value zero and variance σ^2, then the expected squared error at point x_i is

$$ESE\ (x_i|k) = (f(x_i) - \frac{1}{k} \sum_{i-k/2}^{i+k/2} f(x_j)\ )^2 + \frac{1}{k}\sigma^2. \qquad (4)$$

Increasing the span will (if $d^2f/dx^2 \neq 0$) increase the first term, the bias component of the estimation error and decrease the second term, the variance component; decreasing the span will have the opposite effect. The span should be chosen such that both compnents of the error are reasonably balanced. Stated more geometrically, a larger span makes the smooth appear less wiggly by more strongly damping high frequency components of the series (x_i, y_i).

We have, so far, said nothing useful on how to choose the span in practice. The advice given above on balancing bias and variance is not very helpful because both f and the variance of the random error are unknown.

One can estimate the optimal span value in a particular situation as that value that minimizes an estimate for the integrated squared error

$$I^2(k) = \int ESE(x|k)\, dF(x).$$

Using the average squared residual of the data from the smooth

$$\hat{I}^2(k) = \frac{1}{n} \sum_{i=1}^{n} [y_i - s(x_i|k)]^2$$

for this purpose is not appropriate since this is always minimized by the span value k=1. A better estimate is provided by a method referred to as "cross-validation" (M. Stone, 1974) or "predictive sample reuse" (Geisser, 1975) Each observation is in turn deleted and the value of the smooth $s_{(i)}(x_i|k)$ at x_i is calculated from the other n-1 observations. The cross-validated estimate of the integrated square error is

$$\hat{I}^2_{cv}(k) = \frac{1}{n} \sum_{i=1}^{n} [y_i - s_{(i)}(x_i|k)]^2. \qquad (5)$$

Clearly, $E[\hat{I}^2_{cv}]$ equals the expected squared error obtained by applying the procedure to a sample of n-1 observations from the same distribution. The cross-validated estimate for the optimal span value is taken to be the value k_{cv} that minimizes (5),

$$k_{cv} = \min^{-1}_{0<k\leq N} \hat{I}^2_{cv}(k).$$

Model selection through cross-validation has been remarkably successful in a wide variety of situations (see M. Stone, 1974, Geisser, 1975, Craven and Wahba, 1957, C. Stone, 1981).

For the moving average smoothers discussed in the previous section, the deleted smooth estimates $s_{(i)}(x_i|k)$ are especially easy to compute; each observation is simply deleted from the neighborhood used to compute its local straight line fit. Again, the use of updating formulas makes this computation very rapid. As one moves from observation i to i+1, exactly two observations enter the neighborhood (i and i+k/2+1) and exactly two leave it (i+1 and i-k/2). The (deleted) residual squared

$$r^2_{(i)} = [y_i - s_{(i)}(x_i|k)]^2 \tag{6}$$

is computed for each observation and then averaged over all observations,

$$\hat{I}^2_{cv}(k) = \frac{1}{n}\sum_{i=1}^{n} r^2_{(i)}. \tag{7}$$

For small to moderate changes in k, $\hat{I}^2_{cv}(k)$ changes very little so that it is adequate to evaluate it for several (4 to 7) discrete values of k in the range $[0 < k/n \leq 1]$. The value of k corresponding to the smallest of these values is then used. This can be accomplished by maintaining several running average smoothers - one for each span value - in the pass over the data, thus keeping the computational cost linear in n.

So far, we have been assuming that the span is constant over the whole range of predictor values. This is not optimal if either the variance of the random component or the second derivative of the underlying function f change over the range of predictor values. A local increase in error variance would call for an increase in span, whereas an increase in second derivative of f would require a decrease. It is, therefore, desirable to allow the span value to adapt to these changing conditions. This requires that the optimal span value be choosen locally rather than choosing a single global value. Again, the form of moving average smoothers make this especially easy; the (deleted) residual squared (6) -for each of the several k values- is averaged locally in a neighborhood of each observation

$$\hat{I}^2_{cv}(k;x_i) = \frac{1}{L} \sum_{\ell=i-L/2}^{i+L/2} r^2_{(\ell)}(x_\ell|k) \tag{8}$$

rather than globally over all observations (7). Note that (8) also has the form of a simple moving average smoother and can therefore be computed rapidly through the use of updating formulas. The value that minimizes (8)

$$\hat{k}_{cv}(x_i) = \min_{0<k\leq N}^{-1} \hat{I}^2_{cv}(k;x_i) \tag{9}$$

is the span value used for the ith observation.

Most often the shape of $\hat{I}^2_{cv}(k;x_i)$ near its minimum value is shallow and asymmetric, increasing more slowly in the direction of smaller k values. Variability in the estimate $\hat{I}^2_{cv}$, therefore, causes $\hat{k}_{cv}$ to be highly variable and biased toward smaller values. Although this has little effect on the quality of the resulting smooth in terms of expected squared error (ESE), it does effect its aesthetic quality since, for comparable ESE, the less smooth solution tends to be selected. This can be remedied by forcing the procedure to take the smoothest solution in these circumstances. Specifically, the largest span value $\hat{k}^*_{cv}$ for which

$$\hat{I}^2_{cv}(k^*_{cv};x_i) \le (1+\alpha) \min_{0<k\le N} \hat{I}^2_{cv}(k;x_i) \qquad (10)$$

is used for the ith observation. Here α loosely controls an upper limit on the fraction of ESE that is to be sacrificed for the goal of smoothness. Values in the range $0.05 \le \alpha \le 0.2$ are reasonable.

Since the optimal span value is estimated separately for each observation, its size can vary substantially over the range of predictor values. However, since for close abscissa values the neighborhoods overlap considerably, this variation is constrained to be smooth. The degree of smoothness is controlled by the parameter L (8) which can be regarded as a span for smoothing the (deleted) residuals squared from the original smooths. As with the original smoother, its optimal value can be estimated by cross-validation. To the extent that the variation of the second derivative of f or the variation in the random component is comparable to the variation of f itself, this second level of cross-validation may be beneficial. Again, updating formulas make this relatively inexpensive. However, in most circumstances choosing a nominal value for L (0.2n to 0.3n) is adequate.

It is important to note that using cross-validated residuals as a basis for choosing span value is highly sensitive to lack of independence among the ϵ_i (2) as ordered on x. If there is a large positive (negative) correlation among observations with similar x values, substantial under (over) estimates will result. In situations where a high degree of auto-correlation is suspected, these span selection procedures should be used with caution.

Figure 1 illustrates the application of this smoothing algorithm to an artificial data set. (A FORTRAN subroutine implementing this algorithm is listed in Appendix 1.) The data for this example consists of N=200 pairs (x_i, y_i) with the x_i drawn randomly (iid) from a uniform distribution in the interval [0,1]. The y_i are obtained from

$$y_i = \sin[2\pi(1-x_i)^2] + x_i\epsilon_i$$

with the ϵ_i iid standard normal. The parameter ALPHA [α in (10)] was set to 0.1 and RESPAN [L/n in (8)] was set to 0.25 (see Appendix 1). This example simulates a situation in which both the curvature of f decreases and the variance of the random component increases with increasing x. Figure 1a is a scatterplot of the simulated data. Figure 1b also shows this scatterplot, but with the resulting smooth superimposed. The height of the curve near the bottom indicates the span value chosen at each x. The span is seen to increase with x to account for the increasing noise, as well as to take advantage of the decreasing curvature of f. (For X > 0.7, the span has reached the largest value provided in the program, K/N = 0.7.) Figure 1c is the same as Figure 1b except that the curve $Y=f(x)=\sin[2\pi(1-x)^2]$ is superimposed for reference. The resulting smooth is seen to estimate the underlying f reasonably well. Note that for a linear function $y=\alpha x+b$ (zero curvature) the smoother will tend to use a constant (maximum) span value regardless of (the variation of) the amplitude of the noise.

5. Reducing Computation by Binning

In the previous sections, we have described a fairly intricate scatterplot smoother. As an essential building block of projection pursuit regression (Friedman and Stuetzle, 1981), it has performed well. In this context, the smoother is applied to the full data set many times in a single run. In order for such a procedure to be computationally feasible for large data sets, it is necessary that the smoother be as fast as possible. One possibility to increase speed is by binning. Denote the observations for one particular scatterplot by $(x_1,y_1)\ldots(x_n,y_n)$. We assume here that the observations already have been sorted so that the x_i are in increasing order. Choose a bin size, say m, and define new data points $(u_1,v_1)\ldots(u_{n/m},v_{n/m})$ by

$$u_i = \text{mean } (x_{(i-1)m+1}\ldots x_{im});$$

$$v_i = \text{mean } (y_{(i-1)m+1}\ldots y_{im}).$$

Then apply the smoother described above to the $(u_1,v_1)\ldots(u_{n/m},v_{n/m})$. The smooth for predictor values x_j not among $u_1\ldots u_{n/m}$ can be obtained by linear interpolation or, at the boundaries, by extrapolation.

The computing time for the smoother grows linearly in the number of observations, and so binning reduces the run time of the smoother roughly by a factor of m.

Figure 2 shows the results of applying the smoother to a sample of n=500 observations generated from the same model as the data shown in Figure 1, with the results of applying the binning procedure with m=5 superimposed. The quality of the smooth is seen to suffer very little while the computation has been substantially reduced.

6. Resistance

As for all data analytic procedures, it is highly desirable for a scatterplot smoother to be resistant against occasional outliers in the data. (All our analysis is conditional on the observed predictor values; outlier thus means outlier in response.) The smoother described in Sections 4 and 5 clearly is not resistant. One way to overcome this limitation is to first screen the data with a rejection rule identifying outliers, and then apply the smoother to the cleaned data set.

We suggest a rejection rule based on running medians. A running median smoother with span k is defined by

$$s(x_i) = \mathrm{med}\ (y_i-k/2 \ldots y_i+k/2)$$

The ends of the sequence must be treated specially, most simply by replicating the outermost values defined above. The rejection rule makes five passes over the data:

1) Compute a running median smooth s.

2) Replace $s(x_i)$ by $s^*(x_i)$ obtained by linearly interpolating between $(x_{i-1}, s(x_{i-1}))$ and $(x_{i+1}, s(x_{i+1}))$. The purpose of this step is to ensure a more realistic estimate of spread in steps (3) and (4) for monotone (sub)sequences, which are exactly reproduced by a running median.

3) Smooth the absolute residuals $|r_i| = |y_i - s^*(x_i)|$ by a running median and obtain a sequence $v_1 \ldots v_n$ of local spread estimates v^*_i.

4) Smooth the sequence of local spread estimates by a moving

average with span f^*n and obtain smoothed spreads v^*_i. This makes the spread estimates more stable. The same effect could be achieved by increasing the span of the running medians in Step (3); however, this would be more expensive computationally. In the code given in Appendix 2, the constant f is set to 0.3.

5) Flag all observations for which $|r_i| \geq c \cdot v^*_i$. In the code, c is set to 4.5.

Some details related to the treatment of ties have been omitted. A FORTRAN subroutine implementing this algorithm is listed in Appendix 2.

The span for the running medians in Steps (1) and (3) is chosen to be increasing with the sample size n (see Table 2). A motivation for our particular choices is given in Chapter 7. We use the same span in both steps, although there is no inherent need to do so.

Figure 3a shows the result of applying the rejection rule to an artificial data set. The true underlying function is a sine wave. The predictors are uniformly distributed in $[0,2\pi]$; the random errors are Gaussian with standard deviation 0.3. Outliers occur with probability 0.2; they were generated by adding a Gaussian with standard deviation 2.4 to the original observation. Observations flagged as outliers by the rejection rule are marked by squares. Figure 3b shows the results of applying the rejection rule to a real data set.

7. Resistance of Running Medians

The choice of span k for the running medians in Steps (1) and (3) gives rise to rather interesting questions. Somewhat vaguely stated, the rejection rule will be able to detect extreme outliers as long as these running medians do not break down. We will now define precisely how we measure the degree of resistance of a smoother, and give results on the dependence of the resistance of a running median smoother on the span.

Assume we want to smooth a sequence of length n. Responses can be either "good" or "bad", that is, good observations or outliers. We define random variables $b_i \ldots b_n$ by $b_i = 0$ if y_i is good, $b_i = 1$ if y_i is an outlier. Assume that $\mathrm{Prob}(b_i = 1) = p$ and that the b_i are independent. (As noted by Mallows (1980), the latter assumption might not always be realistic; outliers in time series sometimes come in bursts.) A smoothed value s_i is called bad if it can be made arbitrarily large by suitable choice of the reponse values for the bad observations. A smoother is said to suffer a breakdown if one or more of the smoothed values s_i are bad. The probability that this happens under the above model for the b is called the breakdown probability of the smoother. It will generally depend on p and n. A smoother with breakdown probability $(1-p)^n$ is called nonresistant. (For a different definition of breakdown probability, see Mallows (1980).)

We will now devise an approximate formula for the span necessary to guarantee an upper bound on the breakdown probability as a function of n and $p = \mathrm{Prob}(b_i=1)$. For that purpose, we define new random variables $s_1 \ldots s_{n-k+1}$ by

$$s_i = \sum_{i}^{i+k-1} b_j .$$

A running median smoother suffers a breakdown if among any consecutive k observations more than k/2 are bad; i.e., if max $s_i > k/2$. This probability does not seem to be simple to evaluate, but it is easy to obtain an upper bound in terms of the binomial probability Prob(B(k,p) > k/2, using the Bonferroni inequality:

$$\text{Prob}(\max s_i > k/2) \leq (n-k+1)\ \text{Prob}(B(k,p) > k/2).$$

This inequality provides an estimate for the span needed to guarantee a certain upper bound on the breakdown probability for given n and p. Table 1 gives estimates of the necessary span k for breakdown probability bounded by 0.05, and several values of n and p (n=25,50,100,200,400,800; p=0.05,0.1,0.2). For a comparison, we also list the percentage of breakdowns actually observed in thousand randomly generated Bernoulli sequences for the estimated value of k, and the smallest value of k resulting in 50 or fewer breakdowns. The results show that the Bonferroni estimate is close, especially for p=0.05 and p=0.1 where Prob(B(k,p) > k/2) is small; this is in agreement with experience gained in using the Bonferroni inequality in multiple comparisons. The span of the running medians in Steps (1) and (3) of the rejection rule described in Chapter 6 was chosen to guarantee a breakdown probability of less than 0.05 for probability p=0.1 of obtaining an outlier.

Another interesting question is how fast the span k(n) must grow as a function of n with everything else fixed. This question has been answered by P. Erdös and A. Renyi (1970):

<u>Theorem</u>: If $k(n) = c \ln n$ then

$$\text{Prob}\left(\lim_{n \to \infty} \max_{1 \leq i \leq n-k+1} \frac{s_i}{k} = \alpha \right) = 1,$$

where α is related to c via the equation

$$\frac{-1}{c} = \ell n(p(1-p)) + (1-p-\alpha)\ (\ell n \frac{\alpha}{1-\alpha} - \ell n \frac{p}{1-p}),$$

for $\alpha > c$.

This theorem is a special case of Erdös and Renyi's theorem 2. It can be applied to our situation as follows: Choose $\alpha = 1/2 - \epsilon$. Then there exists an n_0 such that for all $n > n_0$ we have max $s_i <$ k/2 for almost all sampling sequences. In addition, Erdös and Renyi show that

- If k grows slower than ln n $(k(n)/\ln n \to 0.)$, then for all but finitely many values of n, max $s_i = k$ for almost all sampling sequences. ("k cannot grow slower than ln n".)
- If k grows faster than ln n $(k(n)/\ln n \to \infty)$, then lim max $s_i = kp$ for almost all sampling sequences; i.e., the strong law of large numbers applies. ("k does not have to grow faster than ln n".)

6. An Updating Algorithm for Running Medians

There is a straightforward way to compute running medians: Obtain the median of each consecutive k-tupel by sorting it. That can be substantially improved upon by making use of the fact that the set of responses defining s_{i+1} is almost the same as the set of responses defining s_i; only $y_{in} = y_{i+k/2+1}$ has to be added, and $y_{out} = y_{i-k/2}$ has to be deleted. The following rules are easy to verify:

- If $y_{in} = s_i$, then $s_{i+1} = s_i$.
- If $y_{in} > s_i$ and $y_{out} > s_i$ or if $y_{in} < s_i$ and $y_{out} < s_i$, then $s_{i+1} = s_i$.

So in the case of random data, we have to do nothing but make these tests half the time.

- If $y_{in} > s_i$ and $y_{out} < s_i$, then let k^+ denote the number of observations in the new span that are bigger than s_i. If $K^+ < k/2$, then $s_{i+1} = s_i$, else s_{i+1} is the smallest observation strictly bigger than s_i. The analog of that is true if $y_{in} < s_i$ and $y_{out} > s_i$.
- If $y_{in} > s_i$ and $y_{out} = s_i$, then define k^+ as above. If $k^+ = k/2$, then s_{i+1} is the smallest observation in the span strictly bigger than s_i; else s_{i+1} is the smallest observation in the span that is bigger than or equal to s_i. The analog of that is true if $y_{in} < s_i$ and $y_{out} = s_i$.

In appendix 2, we give a FORTRAN subroutine that implements the algorithm outlined above.

For random data, the algorithm will take O(nk) operations. It is possible to reduce that to O(n log k) by organizing the observations in the span into a binary tree which is kept balanced as observations are moved in and out (AVL-tree; see Knuth (1973), pp 451). Unfortunately, for the range of k that we have in mind (about 20), log k is not enough smaller than k to compensate for the increased overhead.

9. Discussion

Cleveland (1979) has suggested a scatterplot smoother also based on local linear fits. It differs from the one described in this report mainly in three respects:

- It does not use variable span.
- In the fit of the local straight line determing the smooth $s(x_i)$ for

predictor value x_i, the observations are weighted according to their distance from x_i; observations towards the extremes of the span receive lower weights than observations with predictor values close to x_i. Asymptotic calculations suggest that assigning unequal weights should reduce the error of the curve estimate, but there is no evidence that it makes a substantial difference for sample sizes occurring in practice. It does, however, produce a smoother looking estimate.

- The procedure derives its resistance properties not from data screening with a rejection rule. Instead, each of the local straight lines is fitted, not by least squares, but by a resistant fitting procedure.

Updating formulas cannot be used in this scheme, making it comparatively expensive in terms of computing. To reduce computation, Cleveland suggests evaluating the smooth only for every m-th predictor value. The parameter m here plays a similar role as our bin size; it would be chosen as a fraction of k. We developed our smoothing procedure because variable span is often important, and because the use of updating formulas dramatically reduces computation.

Another class of procedures suggested for smoothing are procedures based on splines. A spline function s of order ℓ with knots at $z_1 \ldots z_k$ is a function satisfying the following two conditions:

- In each of the intervals $(-\infty,z_1),(z_1,z_2)\ldots(z_{k-1},z_k),(z_k,\infty)$, s is a polynomial of degree $\ell - 1$;
- s has $\ell - 2$ continuous derivatives.

One way to use spline functions in scatterplot smoothing is to fit a spline function with knots $z_1 \ldots z_k$ to the data $(x_1,y_1) \ldots (x_n,y_n)$, either by least squares or by some resistant method. The degree of smoothness is determined by the number and position of the knots. A major disadvantage of this method is that k+1 parameters must be chosen: the number and the positions of the knots. Usually some heuristic procedure is used to place the knots once k has been fixed (Jupp, 1978). This leaves the number of knots to be determined. This number plays the role of the span in determining the degree of smoothing. Unfortunately, the output of the smoother can depend on k in a very nonlinear way; it is easy to construct examples where the addition of one more knot substantially decreases the residual sum of squares, whereas further knots hardly make any difference. This makes k more difficult to choose than the span in a local averaging smoother. Furthermore, least squares fit of splines is substantially slower so that choosing k through cross-validation is usually too expensive.

Another way is to use smoothing splines in the sense of Reinsch (1967). A smoothing spline s of order 2ℓ for smoothing parameter λ is the function that minimizes

$$\sum (y_i - f(x_i))^2 + \lambda \int_{x_1}^{x_n} f^{(\ell)2}(x)dx$$

among all functions f with ℓ derivaties. The solution really turns out to be a spline function of order 2ℓ with knots $x_1 \ldots x_n$; the name is thus justified. The larger λ is chosen, the smoother s becomes; thus, λ here plays the role of the span. Computation of the spline for given λ requires the solution of a banded n*n linear system A drawback of the method, as described here, is that it is impossible to obtain an intuitive feeling for the choice of λ in a given example. So, one usually fixes not λ, but the residual sum of squares around the smooth. The corresponding value of λ then has to be found iteratively by repeatedly solving the minimization problem. This substantially increases the necessary amount of computation. Algorithms to determine the optimal λ by cross-validation usually require computation of the singular value decomposition of an nxn matrix; they are expensive and infeasible for sample sizes larger than 200-300.

To summarize, the local averaging smoother described in this report has two desirable properties that set it apart from other scatterplot smoothers: it is very fast to compute and the value of the parameter that controls the amount of smoothing is optimized locally (through cross-validation), allowing it to change over the range of predictor values.

REFERENCES

Cleveland,W.S. (1979). "Robust locally weighted regressio and smoothing scatterplots," J. Amer. Statist. Assoc., 74, 828-836.

Erdös, P. and Renyi, A. (1970). "On a new law of large numbers." J. Anal. Math 23, 103-111.

Craven, P. and Wahba, G. (1979). Smoothing noisy data with spline functions. Estimating the correct degree of smoothing by the method of generalized cross-validation. Numerische Mathematik 31, 317-403.

Friedman, J. H. and Stuetzle,W. (1981). "Projection pursui: regression," J. Amer. Statist. Assoc. 76, 817-823.

Geisser. S. (1975). The predictive sample reuse method with aplications, J. Amer. Statist. Assoc. 74, 153-160.

Jupp, D. L. (1978). "Approximation to data by splines with free knots," SIAM J. Numer. Anal. 15, 328-343.

Knuth,D.E. (1973). "The art of computer programming," Vol. 3, "Sorting and searching."

Mallows,C.L. (1980). "Some theory of nonlinear smoothers." Ann. Stat. 8, 695-715.

Reinsch, C.H. (1967). "Smoothing by spline functions " Numer. Math. 10, 177-183

Stone, C.J. (1981). Admissible selection of an accurate and parsimonious normal linear regression model. Ann. Stat. 9, 475-485.

Stone, H.M. (1974). "Cross-validatory choice and assessment of statistical predictions." J. Roy. Statist. Soc. B-36, 111-147.

APPENDIX I

The following is a complete listing of a FORTRAN subroutine implementing the smoothing procedure described in this paper.

```
      SUBROUTINE SUPSMU (N,X,Y,W,IPER,ALPHA,RESPAN,IBIN,SMO,SC)
C------------------------------------------------------------------------
C
C SUPER SMOOTHER (FRIEDMAN AND STUETZLE, 1982).
C
C CODED BY: J. H. FRIEDMAN AND W. STUETZLE
C           DEPARTMENT OF STATISTICS AND
C           STANFORD LINEAR ACCELERATOR CENTER
C           STANFORD UNIVERSITY
C           STANFORD CA. 94305
C
C INPUT:
C    N : NUMBER OF OBSERVATIONS (X,Y - PAIRS).
C    X(N) : ORDERED ABSCISSA VALUES.
C    Y(N) : CORRESPONDING ORDINATE (RESPONSE) VALUES.
C    W(N) : (OPTIONAL) WEIGHT FOR EACH (X,Y) OBSERVATION.
C       W < 0 0 => ALL OBSERVATIONS HAVE EQUAL WEIGHT.
C    IPER : PERIODIC VARIABLE FLAG.
C       IPER=1 => X IS ORDINARY ORDERED VARIABLE.
C       IPER=2 => X IS A PERIODIC (CIRCULARLY DEFINED) VARIABLE.
C    ALPHA : FRACTIONAL SMOOTHNESS PENALITY ( SEE (10) SECTION 4).
C    RESPAN : FRACTIONAL SPAN FOR RESIDUAL SMOOTHING
C       ( L/N, SEE (8) SECTION 4).
C       RESPAN .LT. 0 => FIXED SPAN SMOOTHER WITH FRACTIONAL
C                             SPAN = ABS(RESPAN).
C    IBIN : BINNING FACTOR (M, SEE SECTION 7).
C OUTPUT:
C   SMO(N) : SMOOTHED ORDINATE (RESPONSE) VALUES.
C SCRATCH:
C   SC(3,N) : INTERNAL WORKING STORAGE.
C
C NOTE:
C   ALPHA=0.1 AND RESPAN=0.25 ARE REASONABLE VALUES. FOR RESPAN > 0
C   SMOOTHER OUTPUT IS COMPLETELY CROSS-VALIDATED; X(I), Y(I), AND
C   W(I) ARE NOT USED IN THE CALCULATION OF SMO(I). THEREFORE,
C   THE AVERAGE SQUARED RESIDUAL
C                   N
C            ASR = SUM  W(I)*(Y(I)-SMO(I))**2
C                  I=1
C    CAN BE USED AS A GOODNESS-OF-FIT MEASURE FOR THE PURPOSE OF
C    SELECTING OPTIMAL VALUES FOR SMOOTHING PARAMETERS BY
C    REPEATED APPLICATION.
C
C    FOR SMALL SAMPLES (N < 40) OR IF THERE ARE SUBSTANTIAL SERIAL
C    CORRELATIONS BETWEEN OBSERATIONS CLOSE IN X - VALUE, THEN
C    A PRESPECIFIED FIXED SPAN SMOOTHER (RESPAN < 0) SHOULD BE
C    USED. REASONABLE SPAN VALUES ARE 0.3 .GE. ABS(RESPAN) .GE. 0.5.
C
C------------------------------------------------------------------------
      DIMENSION X(N),Y(N),W(N),SMO(N),SC(3,N)
      DOUBLE PRECISION SX(5),SY(5),SXX(5),SXY(5),SUM(5),FBW(5)
      DIMENSION IBW(5),RESQUE(5,101),SMOQUE(5,101)
      INTEGER IN,OUT
      DATA IBW1,SPNMAX /3,0.35/
      DATA BIG,EPS /1.0E20,1.0E-03/
      IF (W(1).LT.0.0) GO TO 20
      DO 10 I=1,N
      SC(3,I)=W(I)
10    CONTINUE
      GO TO 40
20    DO 30 I=1,N
      SC(3,I)=1.
30    CONTINUE
40    IF (X(N).GT.X(1)) GO TO 70
      SX(1)=0.0
      FBW(1)=SX(1)
      DO 50 J=1,N
      SX(1)=SX(1)+SC(3,J)*Y(J)
      FBW(1)=FBW(1)+SC(3,J)
50    CONTINUE
      A=SX(1)/FBW(1)
      DO 60 J=1,N
      SMO(J)=A
60    CONTINUE
      RETURN
70    I=N/4
      J=3*I
      SCALE=X(J)-X(I)
80    IF (SCALE.GT.0.0) GO TO 90
      IF (J.LT.N) J=J+1
      IF (I.GT.1) I=I-1
      SCALE=X(J)-X(I)
      GO TO 80
90    VSML=(EPS*SCALE)**2
      IF (IBIN.LE.1) GO TO 110
      NA=0
      SX(1)=0.0
      SY(1)=SX(1)
      FBW(1)=SY(1)
      DO 100 J=1,N
      SX(1)=SX(1)+X(J)*SC(3,J)
      SY(1)=SY(1)+Y(J)*SC(3,J)
      FBW(1)=FBW(1)+SC(3,J)
      IF (MOD(J,IBIN).NE.0) GO TO 100
      NA=NA+1
      SC(1,NA)=SX(1)/FBW(1)
      SC(2,NA)=SY(1)/FBW(1)
      SC(3,NA)=FBW(1)
      SX(1)=0.0
      SY(1)=SX(1)
      FBW(1)=SY(1)
100   CONTINUE
      IF (MOD(N,IBIN).EQ.0) GO TO 130
      NA=NA+1
      SC(1,NA)=SX(1)/FBW(1)
      SC(2 NA)=SY(1)/FBW(1)
      SC(3,NA)=FBW(1)
      GO TO 130
110   NA=N
      DO 120 I=1,N
      SC(1,J)=X(J)
      SC(2,J)=Y(J)
120   CONTINUE
130   IBW(1)=IBW1
      IDELTA=(SPNMAX*NA-IBW(1))/4.0+0.5
      DO 140 I=2,5
      IBW(I)=MIN0(NA/2,IBW(I-1)+IDELTA)
140   CONTINUE
      DO 150 I=1,5
      SX(I)=0.0
      SY(I)=SX(I)
      SXX(I)=SY(I)
      SXY(I)=SXX(I)
      FBW(I)=SXY(I)
      SUM(I)=FBW(I)
      IF (MOD(IBW(I),2).EQ.0) IBW(I)=IBW(I)+1
150   CONTINUE
      IF (RESPAN.GE.0.0) GO TO 160
      IBWS=1
      IBW(1)=0.5*ABS(RESPAN)*NA
      IF (MOD(IBW(1),2).EQ.0) IBW(1)=IBW(1)+1
      IBW(5)=IBW(1)
      GO TO 170
160   IBWS=5
170   IF (IPER.NE.2) GO TO 220
      IT=NA-IBW(5)+1
      IH=IBW(5)-1
      DO 190 J=IT,NA
      DO 180 I=1,IBWS
      IF (J.LT.NA-IBW(I)+1) GO TO 180
      XT=SC(1,J)
      YT=SC(2,J)
      WT=SC(3,J)
      SX(I)=SX(I)+XT*WT
      SY(I)=SY(I)+YT*WT
      SXX(I)=SXX(I)+XT*XT*WT
      SXY(I)=SXY(I)+XT*YT*WT
      FBW(I)=FBW(I)+WT
180   CONTINUE
190   CONTINUE
      DO 210 J=1,IH
      DO 200 I=1,IBWS
      IF (J.GT.IBW(I)-1) GO TO 200
      XT=SC(1,J)
      YT=SC(2,J)
      WT=SC(3,J)
      SX(I)=SX(I)+XT*WT
      SY(I)=SY(I)+YT*WT
      SXX(I)=SXX(I)+XT*XT*WT
      SXY(I)=SXY(I)+XT*YT*WT
      FBW(I)=FBW(I)+WT
200   CONTINUE
210   CONTINUE
      GO TO 250
220   IT=2*IBW(5)-1
```

```
      DO 240 J=1,IT
      DO 230 I=1,IBWS
      IF (J.GT.2*IBW(I)-1) GO TO 230
      XT=SC(1,J)
      YT=SC(2,J)
      WT=SC(3,J)
      SX(I)=SX(I)+XT*WT
      SY(I)=SY(I)+YT*WT
      SXX(I)=SXX(I)+XT*XT*WT
      SXY(I)=SXY(I)+XT*YT*WT
      FBW(I)=FBW(I)+WT
230   CONTINUE
240   CONTINUE
250   KBW=MIN0(101,2*INT(0.5*RESPAN*NA+0.5)+1)
      KBWO2=KBW/2+1
      IH=0
      JT=IH
      JM=JT
      DO 370 J=1,NA
      RESMIN=BIG
      IF (J.LT.KBWO2) GO TO 260
      JT=JT+1
      JM0=JM
      JM=MOD(JM,KBW)+1
260   IH=MOD(IH,KBW)+1
      DO 310 I=1,IBWS
      IF (IBWS.NE.5) GO TO 270
      XT=SC(1,J)
      YT=SC(2,J)
      WT=SC(3,J)
      SX(I)=SX(I)-XT*WT
      SY(I)=SY(I)-YT*WT
      SXX(I)=SXX(I)-XT*XT*WT
      SXY(I)=SXY(I)-XT*YT*WT
      FBW(I)=FBW(I)-WT
270   OUT=J-IBW(I)
      IN=J+IBW(I)-1
      IF ((IPER.NE.2).AND.(OUT.LT.1.OR.IN.GT.NA)) GO TO 280
      IF (OUT.LT.1) OUT=NA+OUT
      IF (IN.GT.NA) IN=IN-NA
      XT=SC(1,OUT)
      YT=SC(2,OUT)
      WT=SC(3,OUT)
      SX(I)=SX(I)-XT*WT
      SY(I)=SY(I)-YT*WT
      SXX(I)=SXX(I)-XT*XT*WT
      SXY(I)=SXY(I)-XT*YT*WT
      FBW(I)=FBW(I)-WT
      XT=SC(1,IN)
      YT=SC(2,IN)
      WT=SC(3,IN)
      SX(I)-SX(I)+XT*WT
      SY(I)=SY(I)+YT*WT
      SXX(I)=SXX(I)+XT*XT*WT
      SXY(I)=SXY(I)+XT*YT*WT
      FBW(I)=FBW(I)+WT
280   D=SXX(I)-SX(I)**2/FBW(I)
      VAR=D/FBW(I)
      A=0.0
      IF (VAR.GT.VSML) A=(SXY(I)-SX(I)*SY(I)/FBW(I))/D
      SM=A*SC(1,J)+(SY(I)-A*SX(I))/FBW(I)
      IF (IBWS.NE.1) GO TO 290
      SMO(J)=SM
      GO TO 320
290   RES=SC(3,J)*(SC(2,J)-SM)**2
      IF (J.GT.KBW) SUM(I)=SUM(I)-RESQUE(I,IH)
      SUM(I)=SUM(I)+RES
      RESQUE(I,IH)=RES
      SMOQUE(I,IH)=SM
      IF (VAR.LT.VSML.AND.I.LT.5) SMOQUE(I,IH)=BIG
      IF (J.LT.KBWO2) GO TO 300
      SUM(I)=SUM(I)-RESQUE(I,JM)
      IF (JT.GT.1) SUM(I)=SUM(I)+RESQUE(I,JMO)
      IF (SUM(I).GT.RESMIN.OR.SMOQUE(I,JM).GE.BIG) GO TO 300
      RESMIN=SUM(I)
      IS=I
300   XT=SC(1,J)
      YT=SC(2,J)
      WT=SC(3,J)
      SX(I)=SX(I)+XT*WT
      SY(I)=SY(I)+YT*WT
      SXX(I)=SXX(I)+XT*XT*WT
      SXY(I)=SXY(I)+XT*YT*WT
      FBW(I)=FBW(I)+WT
310   CONTINUE
320   IF (IBWS.EQ.1) GO TO 370
      IF (J.GE.KBWO2) SMO(JT)=SMOQUE(IS,JM,
      IF (ALPHA.LE.0.0.OR.J.LT.KBWO2.OR.IS.GE.5) GO TO 370
      RESMIN=(1.0+ALPHA)*RESMIN
      I=5
      GO TO 340
330   I=I+(-1)
340   IF ((-1)*((I)-(IS)).GT.0) GO TO 350
      IF (SUM(I).GT.RESMIN) GO TO 330
350   IF (I.GE.5) GO TO 360
      A=(RESMIN-SUM(I))/(SUM(I+1)-SUM(I))
      SMO(JT)=(1.0-A)*SMOQUE(I,JM)+A*SMOQUE(I+1,JM)
      GO TO 370
360   SMO(JT)=SMOQUE(I,JM)
370   CONTINUE
      IF (IBWS.NE.5) GO TO 440
      JT=JT+1
      DO 430 J=JT,NA
      IH=MOD(IH,KBW)+1
      RESMIN=BIG
      JMO=JM
      JM=MOD(JM,KBW)+1
      DO 380 I=1,5
      SUM(I)=SUM(I)-RESQUE(I,IH)+RESQUE(I,JMO)-RESQUE(I,JM)
      IF (SUM(I).GT.RESMIN.OR.SMOQUE(I,JM).GE.BIG) GO TO 380
      RESMIN=SUM(I)
      IS=I
380   CONTINUE
      SMO(J)=SMOQUE(IS,JM)
      IF (ALPHA.LE.0.0.OR.IS.GE.5) GO TO 430
      RESMIN=(1.0+ALPHA)*RESMIN
      I=5
      GO TO 400
390   I=I+(-1)
400   IF ((-1)*((I)-(IS)).GT.0) GO TO 410
      IF (SUM(I).GT.RESMIN) GO TO 390
410   IF (I.GE.5) GO TO 420
      A=(RESMIN-SUM(I))/(SUM(I+1)-SUM(I))
      SMO(J)=(1.0-A)*SMOQUE(I,JM)+A*SMOQUE(I+1,JM)
      GO TO 430
420   SMO(J)=SMOQUE(I,JM)
430   CONTINUE
440   IT=NA-1
      S2=SMO(1)
      IF (IPER.NE.2) GO TO 450
      A=S2
      SMO(1)=0.25*(SMO(NA)+2.0*S2+SMO(2))
      GO TO 460
450   SMO(1)=0.25*(2.0*S2+3.0*SMO(2)-SMO(3))
460   DO 470 J=2,IT
      S1=S2
      S2=SMO(J)
      SMO(J)=0.25*(S1+2.0*S2+SMO(J+1))
470   CONTINUE
      IF (IPER.NE.2) GO TO 480
      SMO(NA)=0.25*(A+2.0*SMO(NA)+S2)
      GO TO 490
480   SMO(NA)=0.25*(2.0*SMO(NA)+3.0*S2-S1)
490   IF (IBIN.LE.1) GO TO 550
      DO 500 I=1,NA
      SC(2,I)=SMO(I)
500   CONTINUE
      XUP=SC(1,1)-1.
      J=0
      DO 540 I=1,N
      XI=X(I)
      IF (XI.LE.XUP) GO TO 530
      J=J+1
      XLOW=SC(1,J)
      XUP=SC(1,J+1)
      YLOW=SC(2,J)
      YUP=SC(2,J+1)
      IF (XLOW.NE.XUP) GO TO 510
      SLOPE=0.
      GO TO 520
510   SLOPE=(YUP-YLOW)/(XUP-XLOW)
520   IF (J+1.EQ.NA) XUP=X(N)
530   SMO(I)=YLOW+(XI-XLOW)*SLOPE
540   CONTINUE
550   J=1
```

```
560  J0=J
     SY(1)=SMO(J)
     IF (W(1).LE.0.0) GO TO 570
     SY(1)=W(J)*SMO(J)
     FBW(1)=W(J)
570  IF (J.GE.N) GO TO 610
580  IF (X(J+1).GT.X(J)) GO TO 610
     J=J+1
     IF (W(J).GT.0.0) GO TO 590
     SY(1)=SY(1)+SMO(J)
     GO TO 600
590  SY(1)=SY(1)+W(J)*SMO(J)
     FBW(1)=FBW(1)+W(J)
600  IF (J.LT.N) GO TO 580
610  IF (J.LE.J0) GO TO 630
     IF (W(1).LE.0.0) FBW(1)=J-J0+1
     SY(1)=SY(1)/FBW(1)
     DO 620 I=J0,J
     SMO(I)=SY(1)
620  CONTINUE
630  J=J+1
     IF (J.LE.N) GO TO 560
     RETURN
     END
```

APPENDIX II

The following is a complete listing of a FORTRAN subroutine implementing the rejection rule described in this paper.

```
      SUBROUTINE REJECT (PRED,RESP,N,WEIGHT,SCRAT)
C-----------------------------------------------------
C
C REJECTION RULE FOR SMOOTHING (FRIEDMAN AND STUETZLE, 1982)
C
C CODED BY: J. H. FRIEDMAN AND W. STUETZLE
C DEPARTMENT OF STATISTICS AND
C STANFORD LINEAR ACCELERATOR CENTER
C STANFORD UNIVERSITY
C STANFORD, CA. 94305
C
C
C INPUT:
C    PRED(N)    :ABSCISSA VALUES IN INCREASING ORDER
C    RESP(N)    :CORRESPONDING ORDINATE (RESPONSE) VALUES
C    N          :NUMBER OF OBSERVATIONS (X,Y-PAIRS)
C
C OUPUT:
C    WEIGHT(N) :REJECTION FLAGS.
C               WEIGHT(I)=0 IF OBSERVATION I IS CONSIDERED AN OUTLIER
C               WEIGHT(I)=1 OTHERWISE
C
C SCRATCH:
C    SCRAT(N,2):INTERNAL WORKING STORAGE
C
C
C NOTE:
C    REJECT USES SUBROUTINE RUNMED (SEE BELOW)
C
C-----------------------------------------------------
      DIMENSION PRED(N),RESP(N),WEIGHT(N),SCRAT(N,2)
      DATA FACT/4.5/
      DATA RELSPA/0.3/
      IF (N.GT.25) GO TO 10
      IBAND=7
      GO TO 50
 10   IF (N.GT.100) GO TO 20
      IBAND=9
      GO TO 50
 20   IF (N.GT.400) GO TO 30
      IBAND=11
      GO TO 50
 30   IF (N.GT.800) GO TO 40
      IBAND=13
      GO TO 50
 40   IBAND=15
 50   CALL RUNMED (RESP,WEIGHT,N,IBAND)
      IFIRST=IBAND/2+1
      ILAST=N-IBAND/2
      DO 60 I=1,IFIRST
      SCRAT(I,1)=WEIGHT(I)
 60   CONTINUE
      DO 70 I=ILAST,N
      SCRAT(I,1)=WEIGHT(I)
 70   CONTINUE
```

```
      DO 90 I=IFIRST,ILAST
      IM1=I-1
      IP1=I+1
      IF (PRED(IM1).NE.PRED(IP1)) GO TO 80
      SCRAT(I,1)=0.5*(WEIGHT(IM1)+WEIGHT(IP1))
      GO TO 90
 80   SCRAT(I,1)=WEIGHT(IM1)+(WEIGHT(IP1)-WEIGHT(IM1))*(PRED(I)-PRED(IM1
     1))/(PRED(IP1)-PRED(IM1))
 90   CONTINUE
      I=0
 100  IF (I.GE.N-1) GO TO 150
      I=I+1
      MO=I
 110  IF (PRED(I+1).GT.PRED(I)) GO TO 120
      I=I+1
      IF (I.LT.N) GO TO 110
 120  IF (I.EQ.MO) GO TO 100
      NTIE=I-MO+1
      R=0.
      DO 130 J=MO,I
      R=R+SCRAT(J,1)
 130  CONTINUE
      R=R/NTIE
      DO 140 J=MO,I
      SCRAT(J,1)=R
 140  CONTINUE
      GO TO 100
 150  DO 160 I=1,N
      WEIGHT(I)=ABS(RESP(I)-SCRAT(I,1))
 160  CONTINUE
      CALL RUNMED (WEIGHT,SCRAT(1,1),N,IBAND)
      IS2=N*RELSPA/2.
      SUM=0.
      DO 170 I=1,IS2
      SUM=SUM+SCRAT(I,1)
 170  CONTINUE
      ISEFF=IS2
      DO 200 I=1,N
      IF (I.GT.N-IS2) GO TO 180
      SUM=SUM+SCRAT(I+IS2,1)
      ISEFF=ISEFF+1
 180  IF (I.LE.IS2+1) GO TO 190
      SUM=SUM-SCRAT(I-IS2-1,1)
      ISEFF=ISEFF-1
 190  SCRAT(I,2)=SUM/ISEFF
 200  CONTINUE
      I=0
 210  IF (I.GE.N-1) GO TO 260
      I=I+1
      MO=I
 220  IF (PRED(I+1).GT.PRED(I)) GO TO 230
      I=I+1
      IF (I.LT.N) GO TO 220
 230  IF (I.EQ.MO) GO TO 210
      NTIE=I-MO+1
      R=0.
      DO 240 J=M0,I
      R=R+SCRAT(J,2)
 240  CONTINUE
      R=R/NTIE
      DO 250 J=M0,I
      SCRAT(J,2)=R
 250  CONTINUE
      GO TO 210
 260  DO 280 I=1,N
      IF (WEIGHT(I).LE.FACT*SCRAT(I,2)) GO TO 270
      WEIGHT(I)=0.
      GO TO 280
 270  WEIGHT(I)=1.
 280  CONTINUE
      RETURN
      END
C---------------------------------------------------------------
C
C
      SUBROUTINE RUNMED (SEQ,SMO,N,IBAND)
C---------------------------------------------------------------
C
C FAST RUNNING MEDIAN FINDER (FRIEDMAN AND STUETZLE, 1982).
C
C CODED BY : J. H. FRIEDMAN AND W. STUETZLE
C            DEPARTMENT OF STATISTICS AND
C            STANFORD LINEAR ACCELERATOR CENTER
C            STANFORD UNIVERSITY
C            STANFORD, CA. 94305
C
C
C INPUT:
C    SEQ(N)   :RESPONSES IN ORDER OF INCREASING PREDICTOR VALUES
C    N        :NUMBER OF OBSERVATIONS
C    IBAND    :SPAN OF RUNNING MEDIANS (HAS TO BE ODD AND <=21)
C
C OUTPUT:
C    SMO(N)   :SMOOTHED RESPONSES
C
C NOTE:
C    THE MAXIMAL SPAN CAN BE INCREASED BY INCREASING THE DIMENSION
C    OF THE ARRAYS SCRAT AND ITAG
C
C-----------------------------------------------------------------
      DIMENSION SEQ(N),SMO(N)
      DIMENSION SCRAT(21),ITAG(21)
      DATA RINF/1.E20/
      DO 10 I=1,IBAND
      SCRAT(I)=SEQ(I)
      ITAG(I)=I
 10   CONTINUE
      RMIN=SCRAT(1)
      IMIN=1
      DO 20 I=2,IBAND
      IF (SCRAT(I).GE.RMIN) GO TO 20
      RMIN=SCRAT(I)
      IMIN=I
20    CONTINUE
      TEMP=SCRAT(1)
      SCRAT(1)=RMIN
      SCRAT(IMIN)=TEMP
      ITAG(1)=IMIN
      ITAG(IMIN)=1
      I=3
      GO TO 40
30    I=I+1
40    IF ((I).GT.(IBAND)) GO TO 60
      IF (SCRAT(I).GE.SCRAT(I-1)) GO TO 30
      TEMP=SCRAT(I)
      ITEMP=ITAG(I)
      J=I
50    SCRAT(J)=SCRAT(J-1)
      ITAG(J)=ITAG(J-1)
      J=J-1
      IF (SCRAT(J-1).GT.TEMP) GO TO 50
      SCRAT(J)=TEMP
      ITAG(J)=ITEMP
      GO TO 30
60    IBAND2=IBAND/2+1
      RMED=SCRAT(IBAND2)
      DO 70 I=1,IBAND2
      SMO(I)=RMED
70    CONTINUE
      IFIRST=2
      ILAST=IBAND+1
      ISMO=IBAND2+1
      TMED=RMED
80    YIN=SEQ(ILAST)
      YOUT=SEQ(IFIRST-1)
      IF (YIN.GE.RMED) GO TO 180
      IF (YOUT.GE.RMED) GO TO 90
      RNEW=RMED
      GO TO 290
90    IF (YOUT.LE.RMED) GO TO 120
      KMINUS=0
      RNEW=-RINF
      DO 110 I=IFIRST,ILAST
      SI=SEQ(I)
      IF (SI.LT.RMED) GO TO 100
      GO TO 110
100   KMINUS=KMINUS+1
      IF (SI.LE.RNEW) GO TO 110
      RNEW=SI
110   CONTINUE
      IF (KMINUS.GE.IBAND2) GO TO 290
      RNEW=RMED
      GO TO 290
120   KMINUS=0
      RTS=-RINF
      RSE=-RINF
      DO 160 I=IFIRST,ILAST
      SI=SEQ(I)
      IF (SI.LE.RMED) GO TO 130
      GO TO 160
130   IF (SI.GE.RMED) GO TO 150
      KMINUS=KMINUS+1
      IF (SI.LE.RTS) GO TO 140
      RTS=SI
140   IF (SI.LE.RSE) GO TO 160
      RSE=SI
      GO TO 160
150   RSE=SI
160   CONTINUE
      IF (KMINUS.NE.IBAND2) GO TO 170
      RNEW=RTS
      GO TO 290
170   RNEW=RSE
      GO TO 290
180   IF (YIN.LE.RMED) GO TO 280
      IF (YOUT.LE.RMED) GO TO 190
      RNEW=RMED
      GO TO 290
190   IF (YOUT.GE.RMED) GO TO 220
      KPLUS=0
      RNEW=RINF
      DO 210 I=IFIRST,ILAST
      SI=SEQ(I)
      IF (SI.GT.RMED) GO TO 200
      GO TO 210
200   KPLUS=KPLUS+1
      IF (SI.GE.RNEW) GO TO 210
      RNEW=SI
210   CONTINUE
      IF (KPLUS.GE.IBAND2) GO TO 290
      RNEW=RMED
      GO TO 290
220   KPLUS=0
      RTB=RINF
      RBE=RI
      DO 260 [illegible]FIRST,ILAST
      SI=SEQ(I)
      IF (SI.GE.RMED) GO TO 230
      GO TO 260
230   IF (SI.LE.RMED) GO TO 250
      KPLUS=KPLUS+1
      IF (SI.GE.RTB) GO TO 240
      RTB=SI
240   IF (SI.GE.RBE) GO TO 260
      RBE=SI
      GO TO 260
250   RBE=SI
260   CONTINUE
      IF (KPLUS.NE.IBAND2) GO TO 270
      RNEW=RTB
```

```
     GO TO 290
270  RNEW=RBE
     GO TO 290
280  RNEW=RMED
290  RMED=RNEW
     SMO(ISMO)=RMED
     IFIRST=IFIRST+1
     ISMO=ISMO+1
     ILAST=ILAST+1
     IF (ILAST.LE.N) GO TO 80
     DO 300 I=ISMO,N
     SMO(I)=RMED
300  CONTINUE
     RETURN
     END
```

Figure Captions

Figure 1a: Two hundred observations (points) drawn from the model $Y=\sin[2\pi(1-X)^2] + X\epsilon$ with ϵ iid standard normal.

Figure 1b: The data of Figure 1a with the computed smooth superimposed. The height of the bottom curve is proportional to the span value employed at the corresponding abscissa value.

Figure 1c: Same as Figure 1b with the addition of the curve $Y=\sin[2\pi(1-X)^2]$

Figure 2: Five hundred observations from the same model as Figure 1, with the computed smooths for both $m=1$ and $m=5$.

Figure 3a: Output of rejection rule applied to artificial data set. Rejected observations are marked by squares.

Figure 3b: Output of rejection rule applied to real data set.

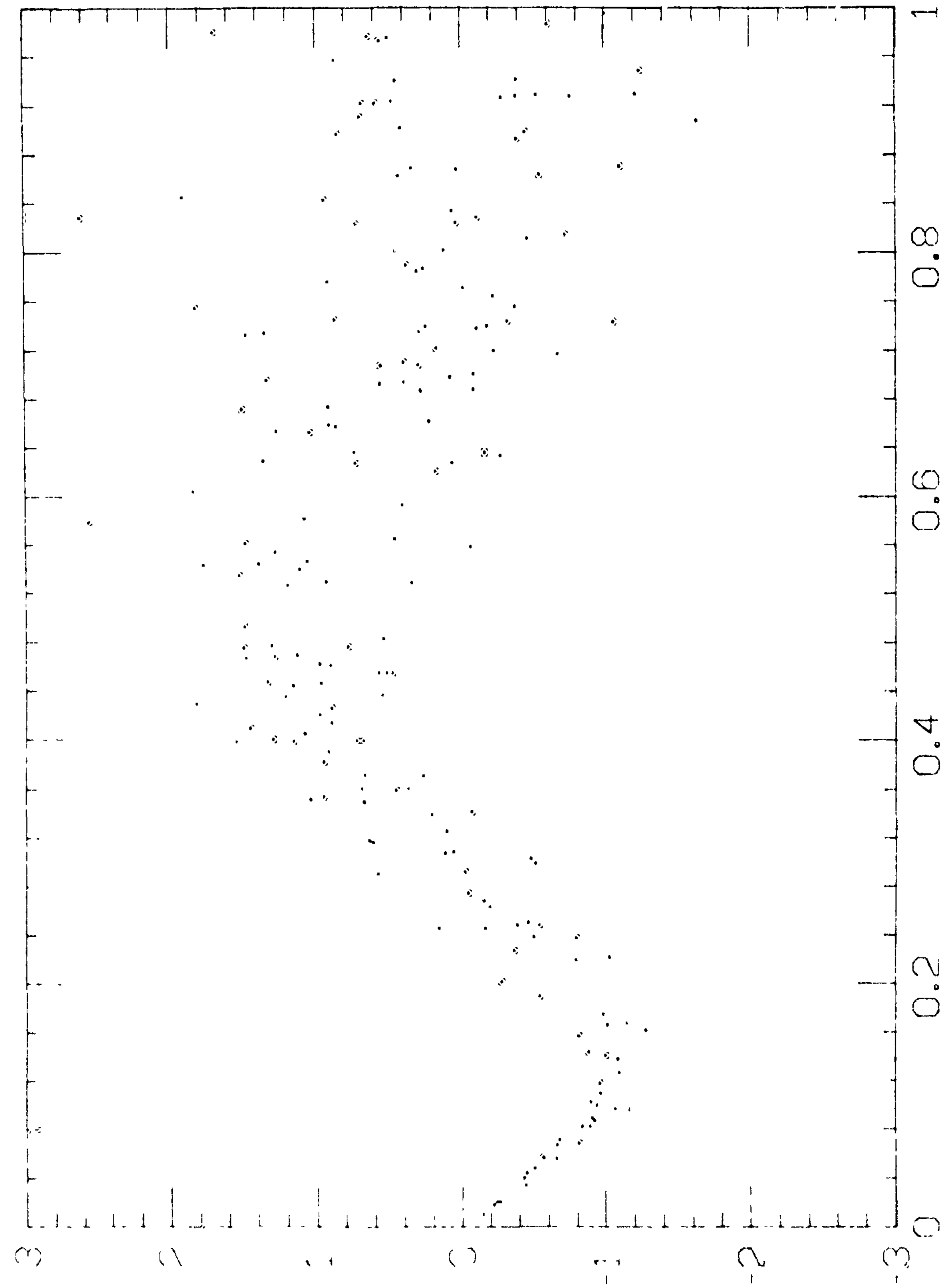

FIGURE 1a

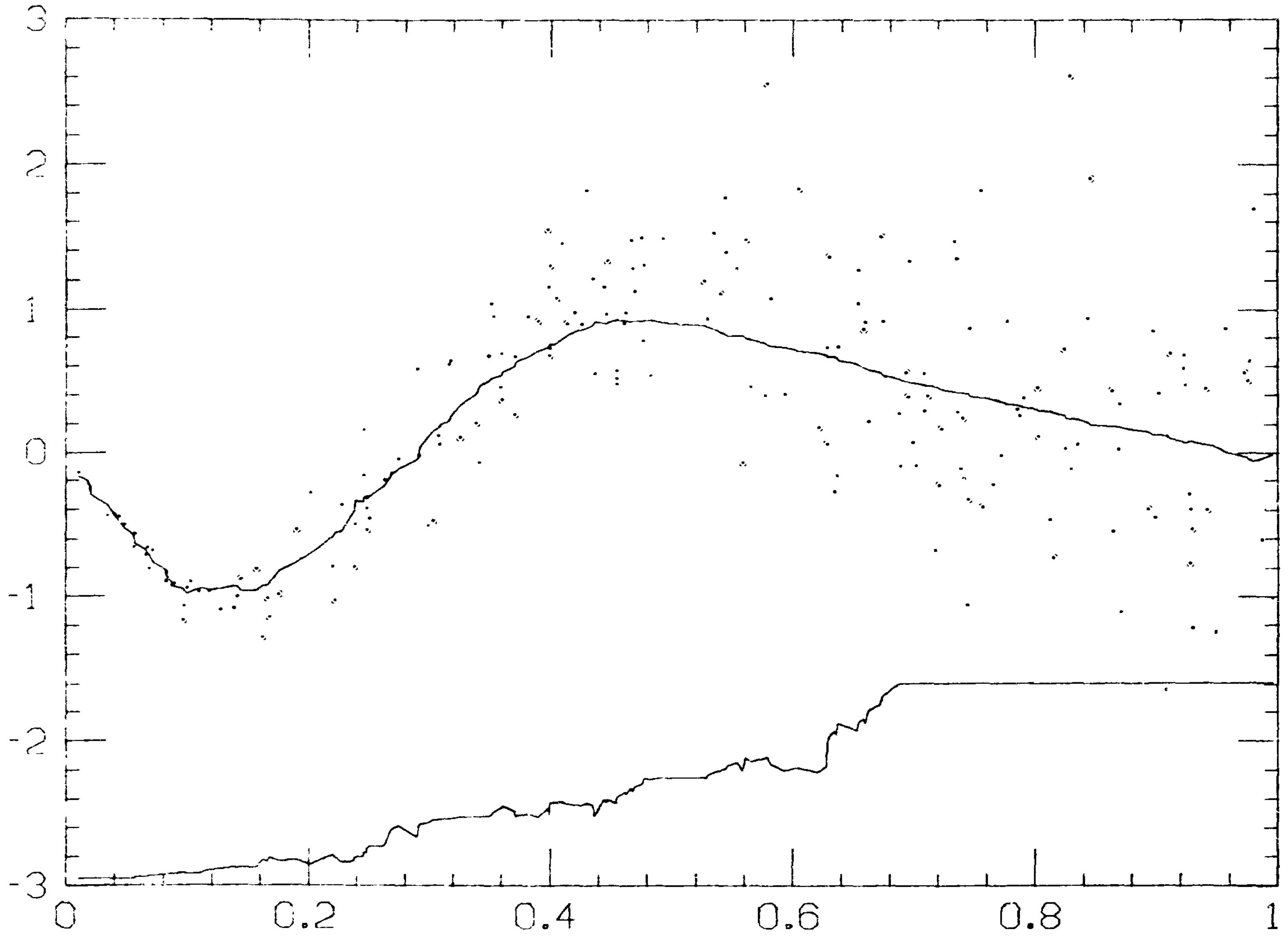

FIGURE 1b

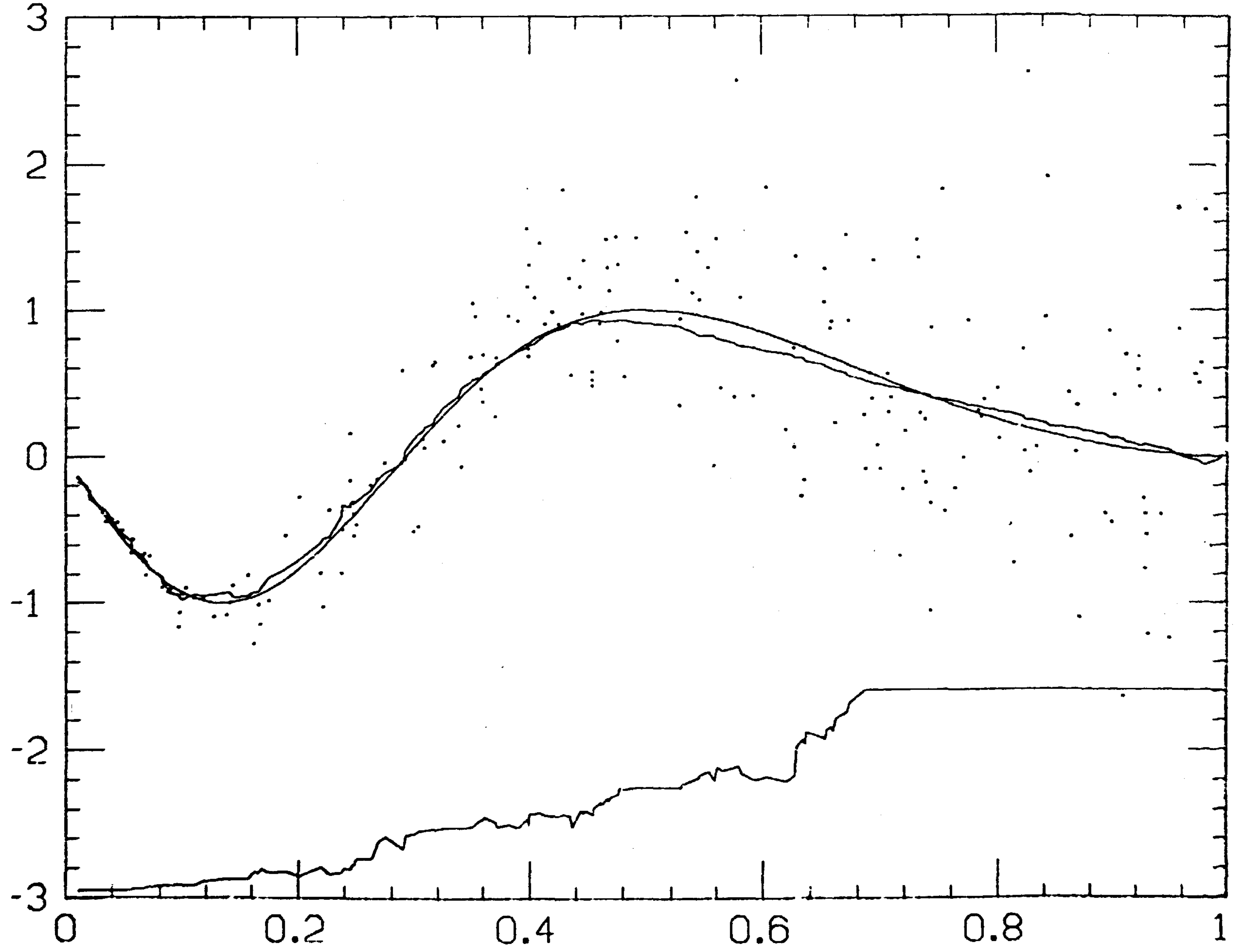

FIGURE 1c

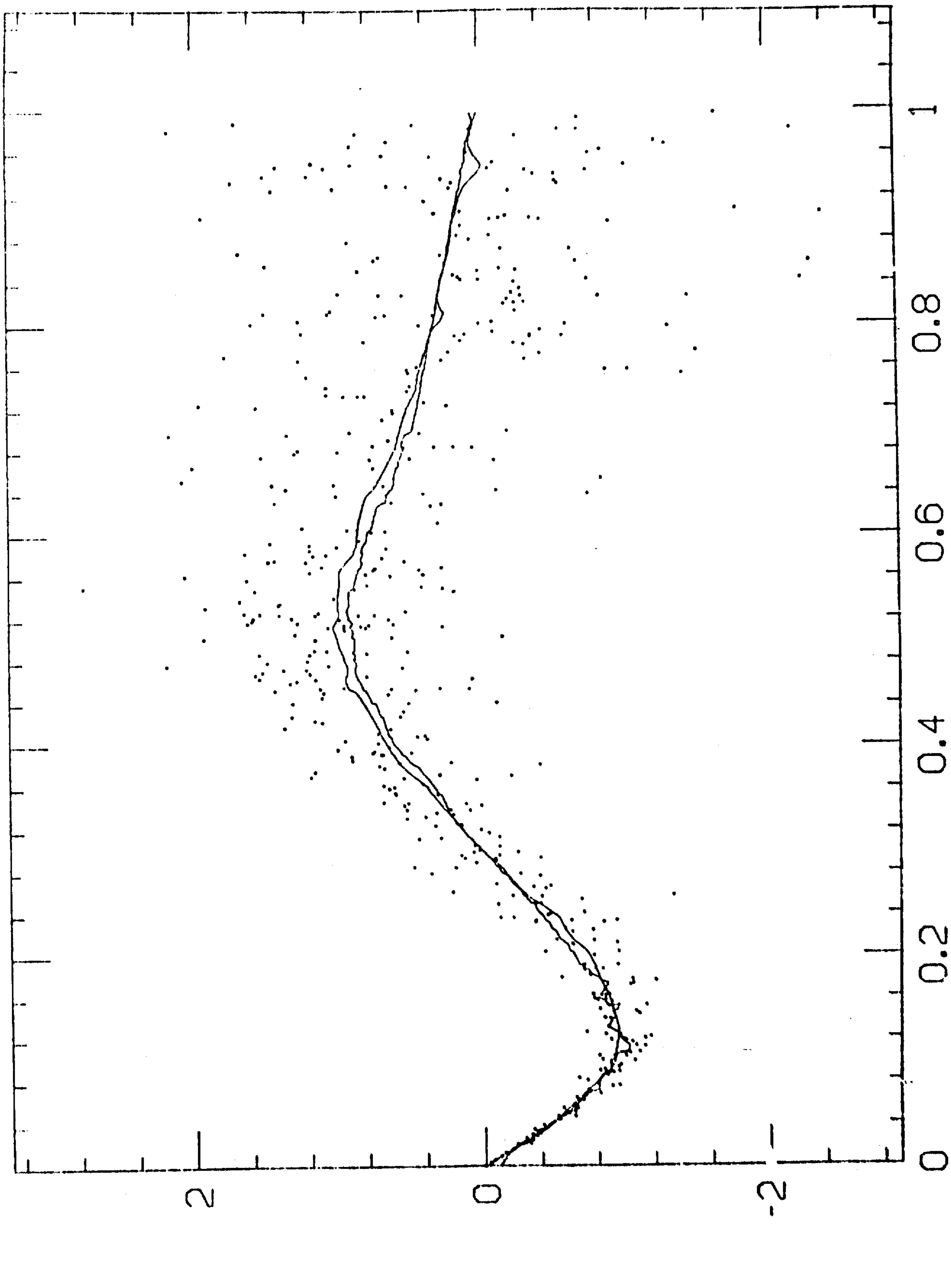

FIGURE 2

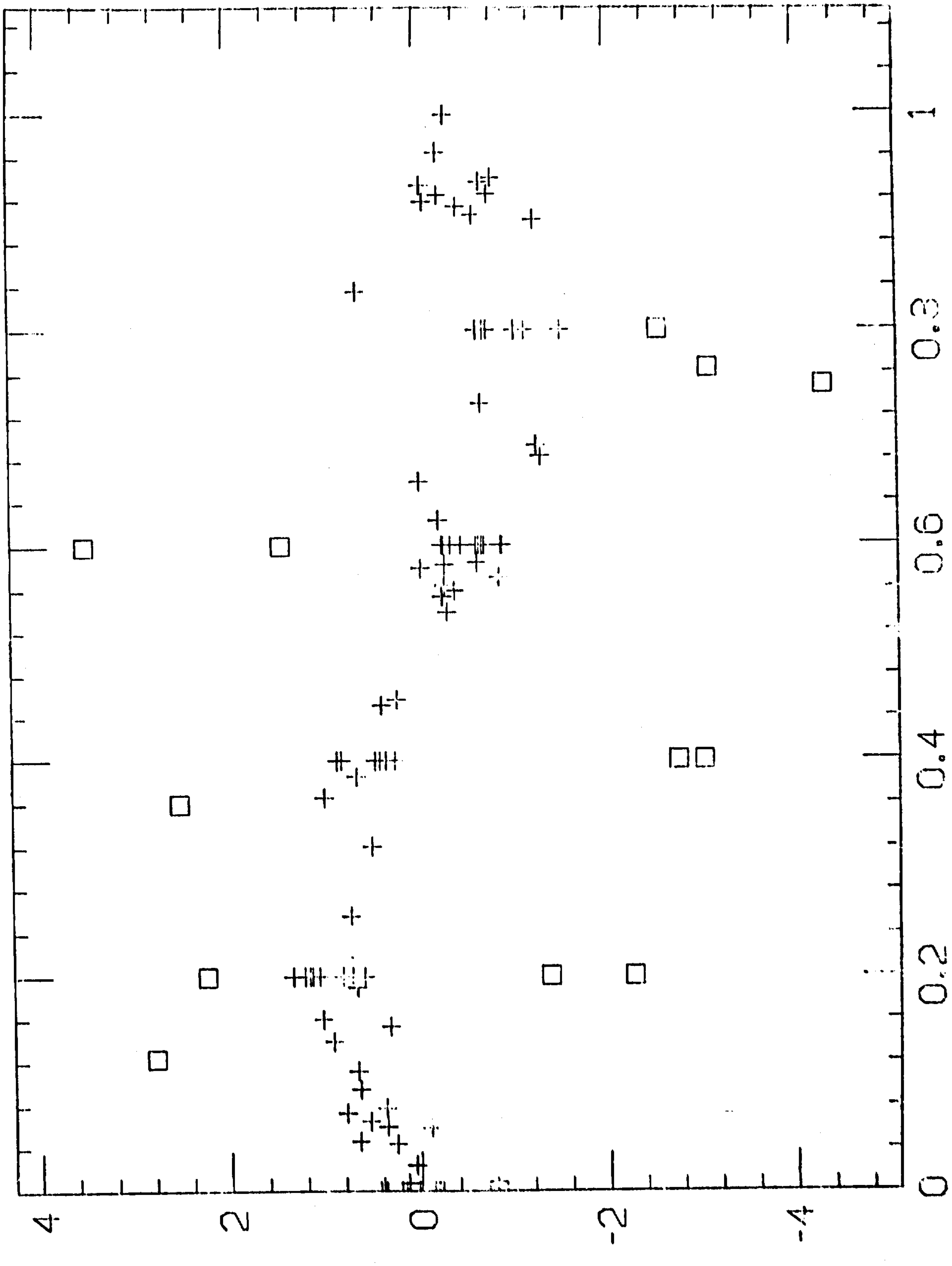

FIGURE 3a

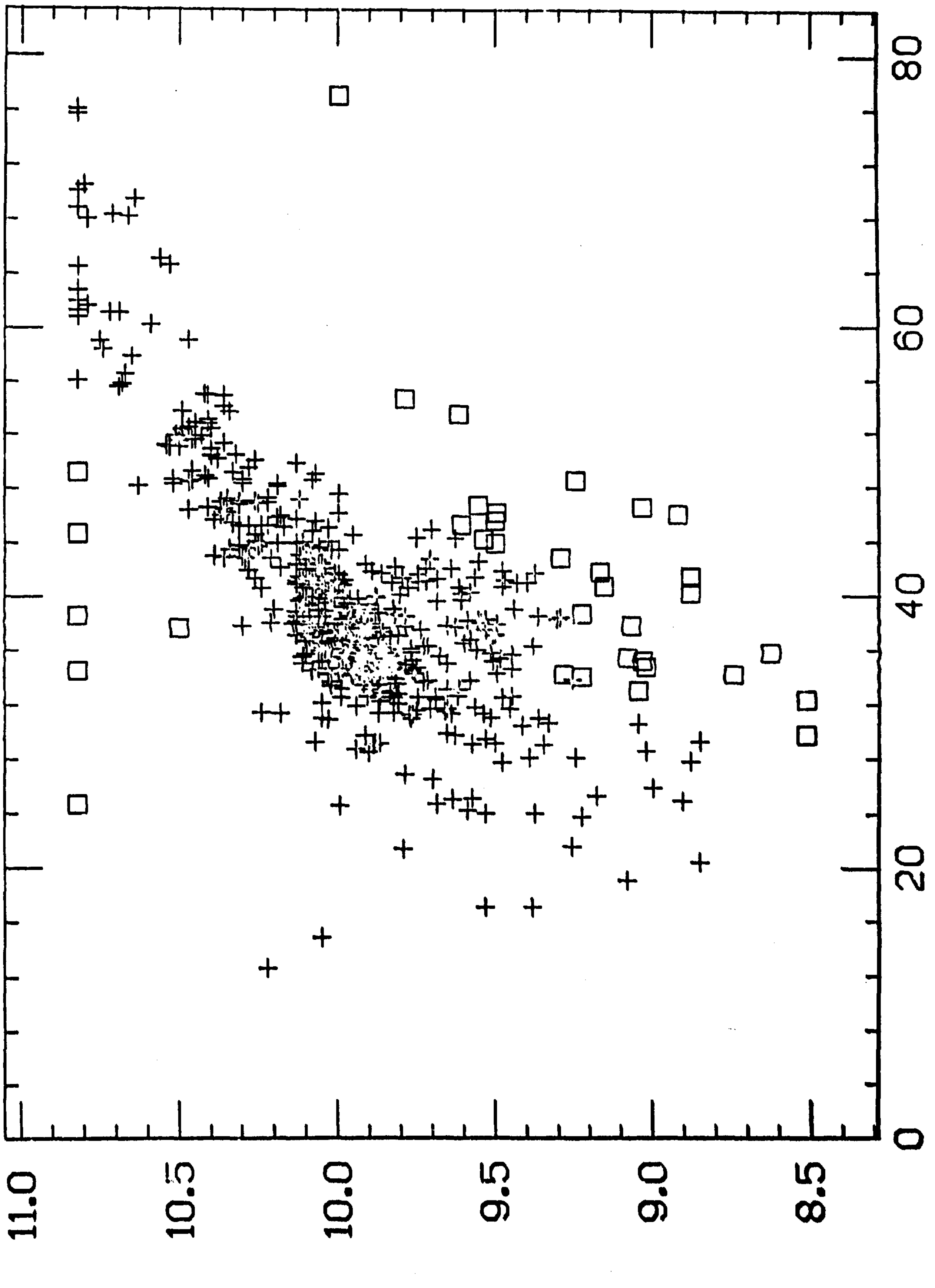

FIGURE 3b

TABLE I

n	P_{bad}	$\bar{K}$	%bds	K
25	0.05	5	1.0	5
50	0.05	7	0.5	5
100	0.05	7	1.0	7
200	0.05	7	2.0	7
400	0.05	9	1.0	7
800	0.05	9	2.0	9
25	0.1	9	0.5	7
50	0.1	9	2.3	9
100	0.1	11	1.4	9
200	0.1	13	0.5	11
400	0.1	13	1.0	11
800	0.1	15	0.0	13
25	0.2	15	1.9	13
50	0.2	21	0.7	15
100	0.2	23	1.4	19
200	0.2	27	1.8	21
400	0.2	31	1.9	27
800	0.2	33	2.1	31

n: length of sequence

P_{bad}: probability of an outlier

$\bar{K}$: Bonferroni estimate of span necessary to guarantee breakdown probability ≤ 0.05

%bds: Percentage of breakdown actually observed in 1000 Monte Carlo trials for span $\bar{K}$.

K: Span necessary to guarantee breakdown probability ≤ 0.05 (estimated from 1000 Monte Carlo trials).

TABLE II

n	K
≤ 25	7
≤100	9
≤400	11
≤800	13
>800	15

n: length of sequence

K: span of running medians in steps (1) and (3) of rejection rule

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER Orion 003	2. GOVT ACCESSION NO. AD-A119 814	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SMOOTHING OF SCATTERPLOTS		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Jerome H. Friedman Werner Stuetzle		8. CONTRACT OR GRANT NUMBER(s) N00014-81-K-0340
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Computer Science Stanford University Stanford, California 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Department of the Navy Office of Naval Research Arlington, Virginia 22217		12. REPORT DATE July 1982
		13. NUMBER OF PAGES 47
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

robustness, smoothing, curve estimation, regression, outlier detection

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A variable span scatterplot smoother based on local linear fits is described. Local cross-validation is used to estimate the optimal span as a function of abscissa value. A rejection rule is suggested to make the smoother resistant against outliers. Computationally efficient algorithms making use of updating formulas and corresponding FORTRAN subroutines are presented.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)